Summary of Activities

The *ns-3* project (http://www.nsnam.org) has completed the fourth year of its initial set of program awards. The project has developed a new discrete-event network simulator oriented towards network research and education, with a special focus on Internet-based systems. The resulting open-source simulator is a community resource for Internet research and education. This report covers project activities for the period April 2010–June 2011 for award CNS-0551378 and is the final report for this project.

The project designed a follow-on successor to the popular *ns–2* simulator (http://www.isi.edu/nsnam/ns). *ns-3* is a rewrite of the entire simulator. During the period of *ns-3* development, *ns–2* is still being maintained.

The project has met all of its stated goals by developing a completely open–source code base with both a high-quality and efficient simulator "core", and a large number of models for various networking components such as TCP/IP, routing protocols, communications links (both wire and wireless), and several networking applications. Further, the *ns-3* simulator has been incorporated into a graduate level course in network simulation (ECE6110) at Georgia Tech. In this class, students use *ns-3* to construct network simulations for a variety of scenarios. The *ns-3* tool has shown to be an excellent vehicle for observing and understanding the behavior of computer networks under a variety of conditions.

There have been a number of software releases for *ns-3* with the most recent being ns-3.12.1. All code included in the releases undergoes a rigorous code review process, and bugs are tracked with an on-line bug tracking web resource known as bugzilla. The *ns-3* documentation consists of a detailed tutorial, an in-depth reference manual, and complete API documentation automatically generated using doxygen.

The project received a set of three additional awards from NSF CISE to develop software frameworks for *ns-3*, including support for scenario generation, simulation automation and management, educational components, and continued software maintenance.

1 Project Organization

ns-3 is organized as an open-source software project, backed by a CISE CRI grant. The NSF team of PIs consists of Tom Henderson (University of Washington) as PI and project lead, and George Riley (Georgia Institute of Technology) and Sumit Roy (University of Washington) as co-PIs. The NSF funding also partially supports staff programmers Craig Dowell (University of Washington) and John Abraham (Georgia Institute of Technology). Finally, several students at the University of Washington and the Georgia Institute of Technology have been supported. The project originally had a fourth co-PI (Dr. Sally Floyd, from the ICSI Center for Internet Research) but Dr. Floyd has taken an early retirement in 2008. The ICIR award funding was transferred to the University of Washington (Dr. Sumit Roy) in 2009, and Tom Henderson committed to complete the work items previously assigned to ICIR.

The University of Washington and INRIA's Planete research group at Sophia Antipolis formalized an "Associated Team" arrangement in 2007. The technical focus of the project is on the integration of *ns-3* with network testbeds such as Rutgers' ORBIT and the European OneLab. As part of this arrangement, Mathieu Lacage at INRIA has worked on the project as a full time software developer, and has been serving as the software lead for the simulation core. He visited the University of Washington for a seven-month period from March–September 2008.

2 Impact

ns-3 has become one of the leading network simulators for networking research. Many statistics point to the increasing usage of the tool. We established an *ns-3* users mailing list on Google Groups, and the list membership grew from 200 to 700 subscribers during this time. The ns-developers mailing list (shared also with *ns–2*) grew from 900 to 1100 subscribers. According to server logs, we average roughly 4000 to 6000 downloads per month of our released software, and we have an uncounted number of people working from the development tree, since it is update more regularly. We now have roughly 260,000 lines of C++ and Python code[1] in the project.

In addition to the aforementioned, a number of independent contributors (mainly students or postdoctoral researchers) from the United States and abroad have started to contribute software to *ns-3*. We expect this number to grow in each coming year, and we view such contributions as essential to the long-term success of the project.

[1] as measured by the cloc script at http://cloc.sourceforge.net

3 ns-3 Software Development

3.1 Overview

The main activity of the project has been software development. *ns-3* is a user-space program that runs on Unix- and Linux-based systems and on Windows (currently via Cygwin). It is written in C++, with a Python scripting interface. The focus is on IPv4 and IPv6-based networks, but other non-IP architectures such as sensors or delay-tolerant networks (DTNs) are to be supported. *ns-3* is meant to be modifiable and extendable by users; some users will be able to use example scripts that are provided, but it is expected that most (research) users will want to either write new scripts or modify or add to the simulator models in some way. Source code distributions are therefore expected to be the preferred means for distributing *ns-3*.

ns-3 contains support for the following:

- Construction of virtual networks (nodes, channels, applications) and support for items such as event schedulers, topology generators, timers, and other objects to support discrete-event network simulation focused on Internet-based and possibly other packet network systems.
- Support for network emulation; the ability for simulator processes to emit and consume real network packets
- Support for animation of network simulations
- Support for tracing, logging, and computing statistics on the simulation output

ns-3 has a modular implementation containing a *core* module supporting generic aspects of the simulator (debugging objects, random number generators, smart pointers, callbacks), and a *simulator* module defining simulation time objects, schedulers, and events. A *common* module defines objects that are independent of specific network architectures, such as generic packets and tracing objects. The *node* module defines abstract base classes for fundamental base objects in the simulator, such as nodes, channels, and network devices. Internet-related models (IP and transport protocols) are found in the *internet-node* module. Specific devices such as WiFi are in the *device* modules. Users may write and link their own modules. All of these modules (the *ns-3* core) are built as a library that is linked at run time with an executable script (the `main()` program).

3.2 Specific Software Modules Developed

Below, we highlight the features added to *ns-3* during this time period.

The fourth release of *ns-3* (ns-3.4) was made in April 2009 and contained the following features:

- *Object Names:* A generic object naming system for *ns-3* objects.
- *Tap Device Support:* Support for bridging *ns-3* ghost nodes to tap devices on the physical host.

Additionally, the April release included enhancements to the random number generator interfaces, calendar queue scheduler, and an extended build system.

The fifth release of *ns-3* (called "ns-3.5") was made in July 2009 and contained the following:

- *802.11 MAC:* 801.11n initial support for A-MSDU frame aggregation, two additional rate control algorithms, and EDCA multiple QOS class support.
- *802.11 PHY:* 802.11b physical layer models, Nakagami propagation loss models, and Radiotap and Prism headers.
- *Random Variables:* Gamma, Erlang, and Zipf distributions
- *IPv4 routing framework:* Fine-grained control over routing policy and protocol composition.

The sixth release of *ns-3* (ns-3.6) was made in October 2009 and contained the following:

- *IPv6:* An initial IPv6 implementation including IPv6 interfaces, network layer, raw socket, static IPv6 routing, ICMPv6, Neighbor Discovery, router advertisements, and a Ping6 application.
- *802.11:* Minstrel rate control algorithm, Athstats helper, 10MHz and 5MHz channel widths, and channel switching support.

- *Wireless mesh:* IEEE 802.11s (Draft 3.0) model including Peering Management Protocol, HWMP, and Forwarding Layer for Meshing (FLAME).
- *NixVector Routing:* Efficient source-routing for static simulation topologies.
- *New test framework:* For regression and validation testing.
- *Flow monitor module:* Ability to trace packet flows.

The seventh release of *ns-3* (ns-3.7) was made in January 2010 and contained the following:

- *Ad hoc On-Demand Distance Vector (AODV):* Routing model for mobile networks.
- *IPv6 extensions and options:* Fragmentation and loose source routing.
- *Animation Interface:* A support object that provides an interface between the low–level packet tracing framework and any arbitrary animation tool.
- *Qt-4 Animator:* A stand–alone animation tool that will display the state of a wired topology, including all nodes, point–to–point links, and packets.
- *Equal-cost multipath:* Routing supoprt for global static routing.

The eighth release of *ns-3* (ns-3.8) was made in May 2010 and contained the following:

- *WiMAX Net Device:* Allow to simulated IEEE 802.16 point to multi-point based networks
- *Distributed simulation:* Support for point-to-point-based simulations for decomposition to multiple machines using Message Passing Interface (MPI) standard.
- *Topology reader:* Allows quick and easy creation of large topologies by reading Inet or Orbis files.
- *Two-ray ground propagation loss model:* Calculates the crossover distance under which the Friis model is used.
- *Tracing support:* Unified tracing across all NetDevice models.

Of the above contributions, the object names, tap device support, IPv4 routing framework, Nix-Vector routing, test framework, animation interface, Qt-4-based animator, MPI-based distributed simulations, and tracing framework were primarily contributed by personnel funded by our NSF project. As primary software maintainers, the developers funded on this project also played a major role in reviewing and integrating the additional contributions (listed above) to the *ns-3* codebase.

The emphasis in the NSF-funded project has been to focus on the core software architecture and APIs, rather than on model development. We have witnessed a large number of models contributed from a number of outside research organizations. For instance, the following extensions to the 802.11 models were developed by outside projects: 802.11e (University of Karlsruhe), 802.11n (University of Florence), 802.11s (Russian Academy of Sciences), 802.11b (Boeing), and 802.11p (iTETRIS EU project).

3.3 Developer Activities

The project has employed one half-time and one full-time developer whose responsibilities range from software model development to debugging, release management, infrastructure maintenance, documentation, merging code, responding to user or developer mail, and outreach. Craig Dowell (University of Washington) developed and maintained the *ns-3* real-time scheduler, emulation support, and an *ns-3* tap device, and led the work on the integration of the CORE network emulator with *ns-3*Ḣe also served as release manager for two releases during this period, and maintained the *ns-3* tutorial. Josh Pelkey (Georgia Institute of Technology) contributed to the Nix-Vector and MPI contributions, maintains the TCP model, and is serving as ns-3.9 release manager. Raj Bhattacharjea (Georgia Institute of Technology) preceded Josh Pelkey as the TCP maintainer and managed the fourth *ns-3* release. John Abraham (also at Georgia Tech) has replaced Josh Pelkey and continues to maintain and support large parts of the *ns-3* code and develop the animation interface.

4 Meetings and Outreach

Tom Henderson and George Riley organized a workshop on *ns-3* at the third annual SIMUTools conference in 2010 (March 15, 2010, Malaga Spain). The website for this event is http://www.wns3.org. The workshop drew roughly 30 people, and an open ns-developers meeting was held the following day (15 people attended). This year's event featured a full slate of invited and accepted short talks from various research groups that have started to use *ns-3* in their research.

George Riley has made several *ns-3* presentations and detailed tutorials, including SIMUTools 2010 main track, Johns Hopkins Applied Physics Lab Technology Colloquium, keynote talk at 2010 Spring Simulation Conference, keynote talk at 2009 Conference on Network Simulation, and an invited talk at the Naval Postgraduate school.

Tom Henderson and Craig Dowell worked with Jeff Ahrenholz and Tom Goff of Boeing Research and Technology on the integration of *ns-3* with Linux network namespaces and with CORE.

The project applied for and was accepted into the Google Summer of Code project in both 2009 and 2010. Three students were funded by Google in 2009, and four in 2010. Each student was paired with a mentor and focuses on a properly-scoped software project for the summer, with the goal of merging the resulting code by the end of the summer.

We know of roughly ten to fifteen refereed papers using *ns-3* for their research, including two in ACM Sigcomm 2009. Many papers about the tool itself have been published; in particular, the best student paper and runner-up to the best overall paper in SIMUTools 2010 Conference were about *ns-3*.